Python

14

Итератор

```
mylist1 = [1, 2, 3]
for i in mylist1:
    print(i)
```

```
mylist2 = [x*x for x in range(3)]
for i in mylist2 :
    print(i)
```

Считывание элементов один за другим - итерирование. Список - итерируемый объект.

Всё, к чему можно применить цикл «for… in...», является итерируемым объектом: списки, строки, файлы

Генератор

```
mygenerator = (x*x  for x in range(3))
for i in mygenerator :
    print(i)
```

Генераторы - итерируемые объекты, которые можно прочитать только один раз. Генераторы не хранят значения в памяти, а генерируют их на лету.

yield

```
def create_generator() :
    mylist = range(3)
    for i in mylist:
        yield i*i


mygenerator = create_generator()
print(mygenerator)
# <generator object createGenerator at 0x000000>
for i in mygenerator:
     print(i)
```

Yield - ключевое слово, которое используется вместо return и возвращает объект генератора.

Генераторы удобно использовать, когда известно что функция вернет большой набор значений, который надо прочитать только один раз.

Когда вызывается функция, код внутри тела функции не исполняется. Функция только возвращает объект-генератор. Код будет вызываться каждый раз, когда for обращается к генератору.

В первый запуск функции, она будет исполняться от начала до того момента, когда она наткнется на yield — тогда она вернет первое значение из цикла. На каждый следующий вызов будет происходить ещё одна итерация написанного вами цикла, возвращаться будет следующее значение — и так пока значения не кончатся.

Генератор считается пустым, как только при исполнении кода функции не встречается yield. Это может случиться из-за конца цикла, или же если не выполняется какое-то из условий «if/else».

next

```
def my_animal_generator():
    yield 'корова'
    for animal in ['кот', 'собака', 'медведь']:
        yield animal
    yield 'кит'

my_generator = my_animal_generator()
print(next(my_generator))
print('---------')
for animal in my_generator:
    print(animal)
    # корова кот собака медведь кит
```

Для обращения к следующему элементу генератора используется функция **next**

Задание 14.01

Создать бесконечный генератор случайных чисел.

Использовать в генераторе временную задержку

from time import sleep

Задание 14.02

Модифицировать генератор, чтобы генератор принимал диапазон случайных чисел и чтобы последующее случайное число лежало в диапазоне смещенном на n.

Пример: a = 1, b = 10, diff = 10

1. 1- 10
2. 11-20
3. …
4. N +10 - M + 10

Создание скриптов

```
import sys

print(sys.argv)

if 'Alex' in sys.argv:
    print('Hello, Alex')
else:
    print('Hello, Guest')
```

При запуске программ на python из терминала с передачей аргументов

python [filename].py arg1 arg2

sys.argv содержит все переданные аргументы. В качестве первого аргумента выступает путь по которому был запущен скрипт

Задание 14.03

Создать скрипт, который при запуске принимает неопределенное количество аргументов и считает сумму тех из них, что являются цифрами.

Пример:

python test.py 1 2 3 4 a b 5 6 --> 21

argparse

```
import sys
import argparse

print(sys.argv)
parser = argparse.ArgumentParser()
parser.add_argument('-fn', '--first-name',
required=True)
parser.add_argument('-ln', '--last-name',
required=True)
parser.add_argument('echo')
args = parser.parse_args()
print(args)
print('First name:', args.first_name)
print('Last name:', args.last_name)
print('echo:', args.echo)
```

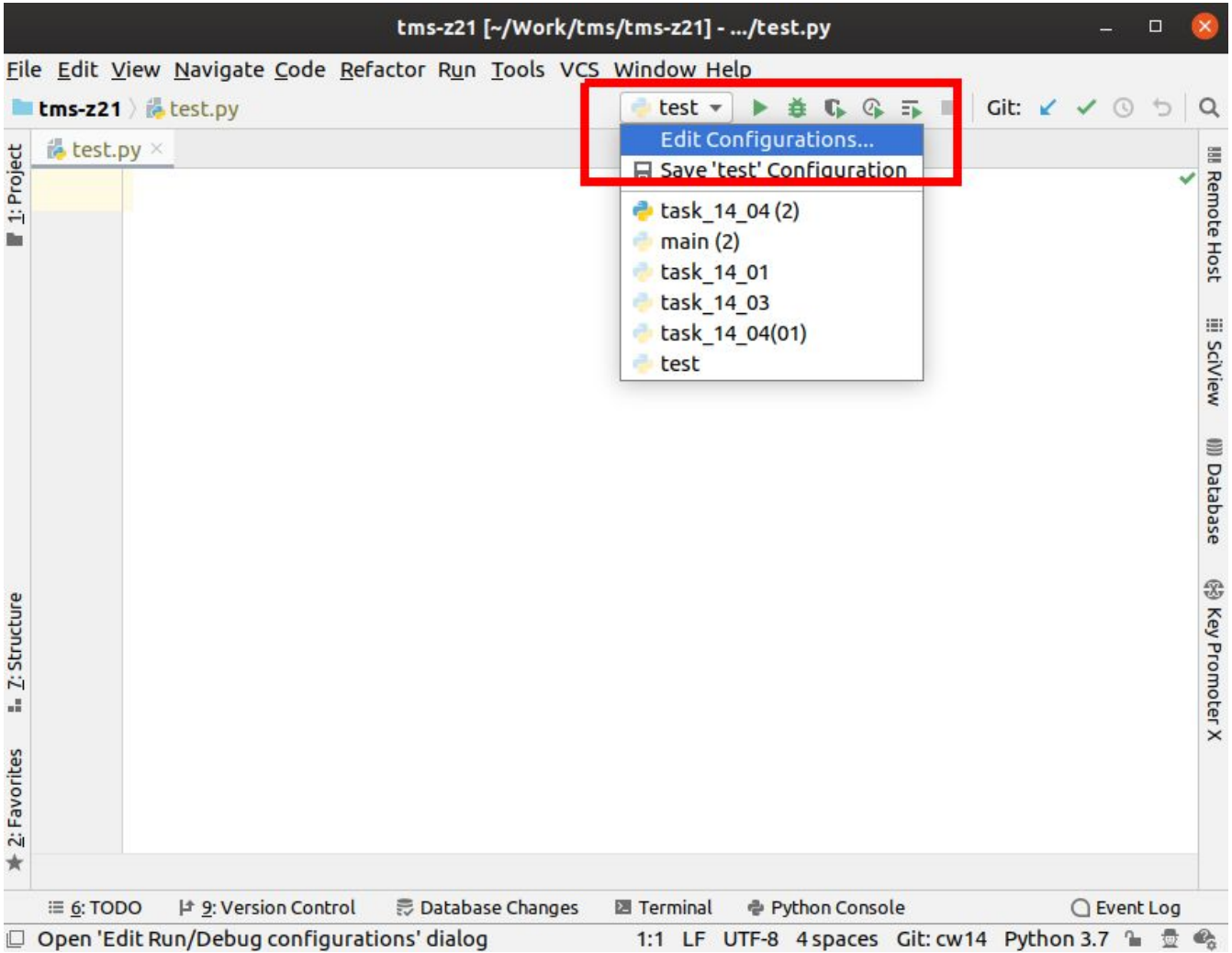
argparse - библиотека упрощающая работу с входными аргументами скрипта.

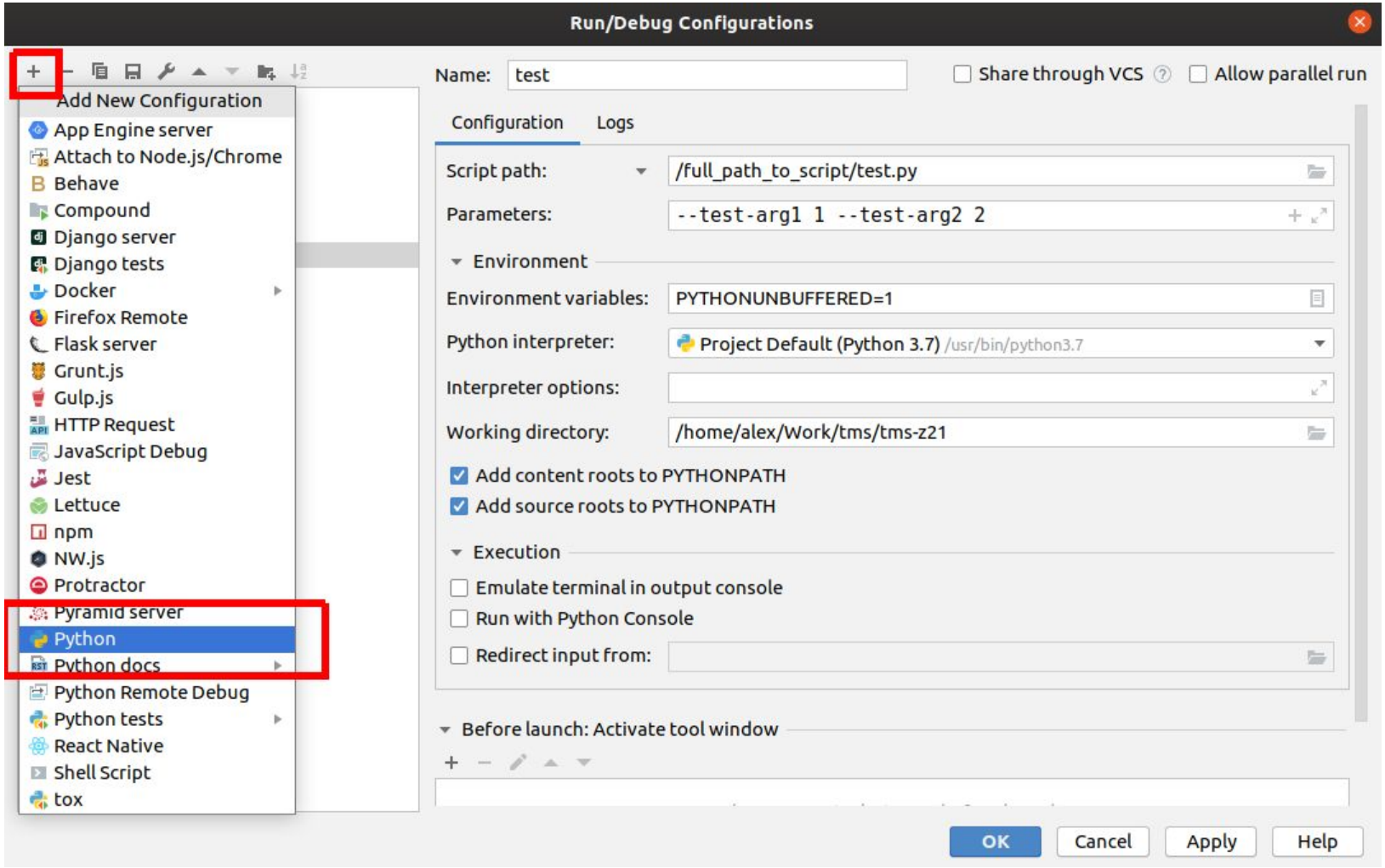
python test.py -fn Alex --last-name Varkalov test
или
python test.py test -fn=Alex --last-name=Varkalov

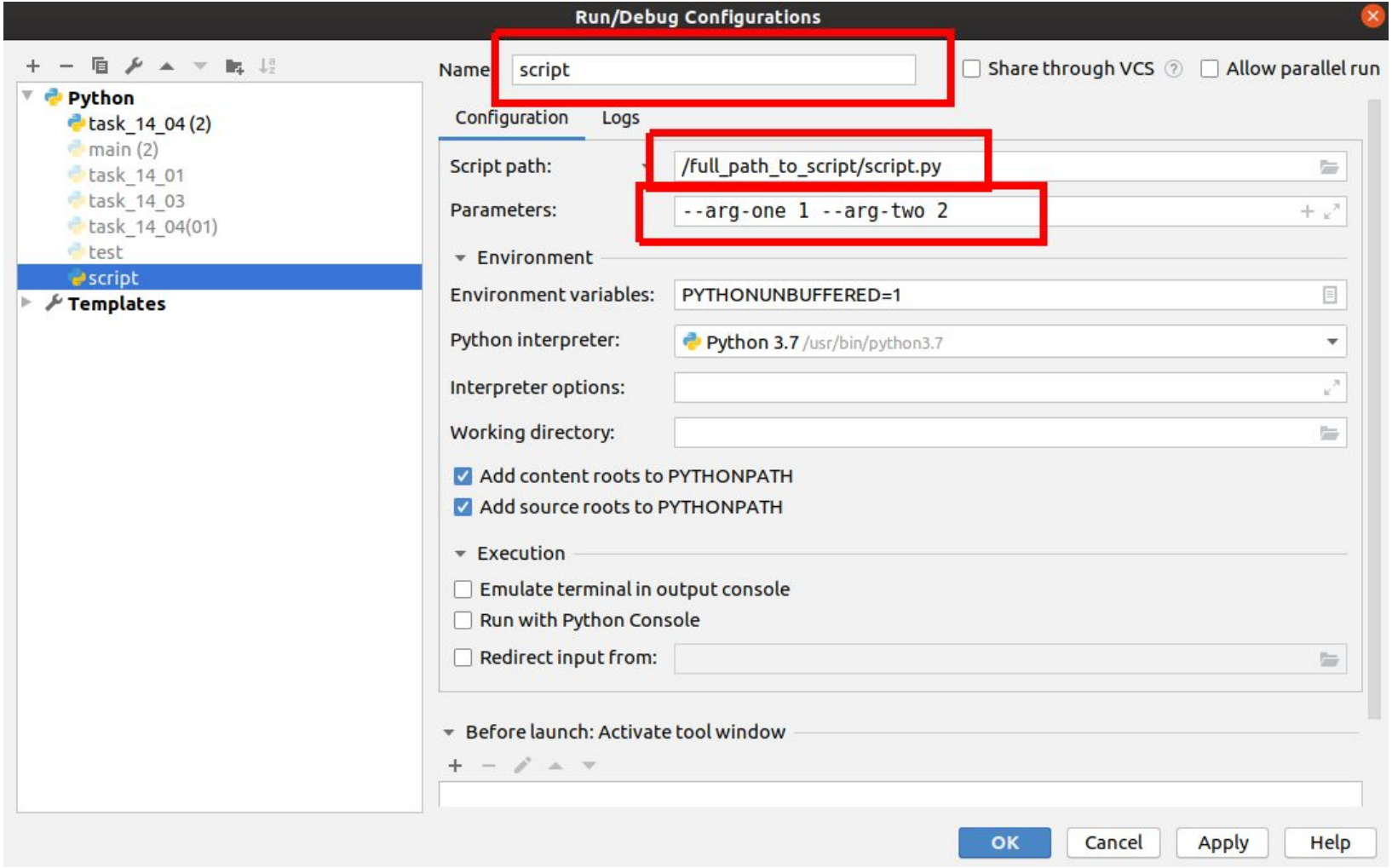
Задание 14.04

Создать скрипт, который принимает имя фамилию и возраст и дописывает их в csv файл

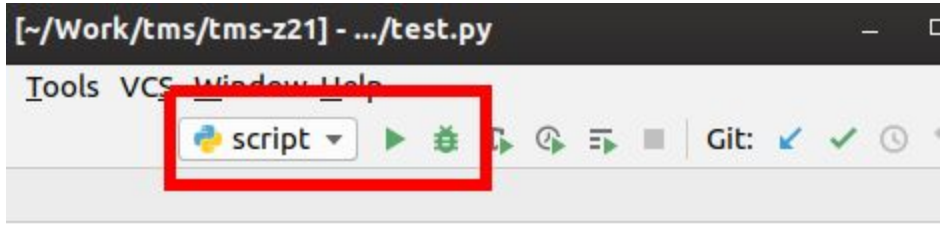
Использование конфигурации

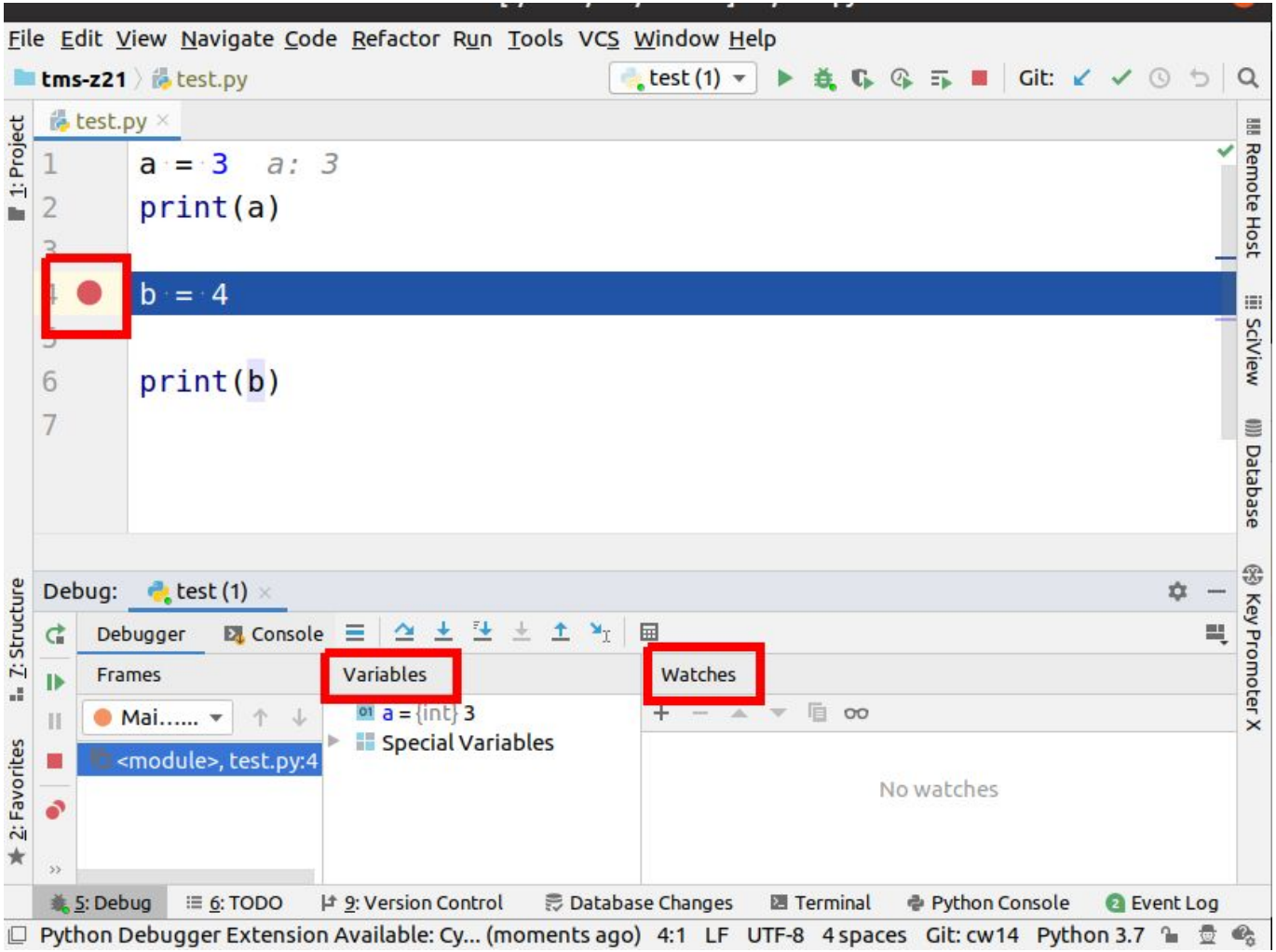

tms-z21 [~/Work/tms/tms-z21] - .../test.py
File Edit View Navigate Code Refactor Run Tools VCS Window Help
tms-z21 test.py
test
Git:
Edit Configurations...
Save 'test' Configuration
task_14_04 (2)
main (2)
task_14_01
task_14_03
task_14_04(01)
test
test.py
1: Project
2: Favorites
7: Structure
Remote Host
SciView
Database
Key Promoter X
6: TODO
9: Version Control
Database Changes
Terminal
Python Console
Event Log
Open 'Edit Run/Debug configurations' dialog
1:1 LF UTF-8 4 spaces Git: cw14 Python 3.7


Run/Debug Configurations
Add New Configuration
App Engine server
Attach to Node.js/Chrome
Behave
Compound
Django server
Django tests
Docker
Firefox Remote
Flask server
Grunt.js
Gulp.js
HTTP Request
JavaScript Debug
Jest
Lettuce
npm
NW.js
Protractor
Pyramid server
Python
Python docs
Python Remote Debug
Python tests
React Native
Shell Script
tox
Name: test
Share through VCS
Allow parallel run
Configuration
Logs
Script path: /full_path_to_script/test.py
Parameters: --test-arg1 1 --test-arg2 2
Environment
Environment variables: PYTHONUNBUFFERED=1
Python interpreter: Project Default (Python 3.7) /usr/bin/python3.7
Interpreter options:
Working directory: /home/alex/Work/tms/tms-z21
Add content roots to PYTHONPATH
Add source roots to PYTHONPATH
Execution
Emulate terminal in output console
Run with Python Console
Redirect input from:
Before launch: Activate tool window
OK
Cancel
Apply
Help


Run/Debug Configurations
Python
task_14_04 (2)
main (2)
task_14_01
task_14_03
task_14_04(01)
test
script
Templates
Name
script
Share through VCS
Allow parallel run
Configuration
Logs
Script path:
/full_path_to_script/script.py
Parameters:
--arg-one 1 --arg-two 2
Environment
Environment variables:
PYTHONUNBUFFERED=1
Python interpreter:
Python 3.7 /usr/bin/python3.7
Interpreter options:
Working directory:
Add content roots to PYTHONPATH
Add source roots to PYTHONPATH
Execution
Emulate terminal in output console
Run with Python Console
Redirect input from:
Before launch: Activate tool window
OK
Cancel
Apply
Help

Использование дебаггера


[~/Work/tms/tms-z21] - .../test.py
Tools VCS Window Help
script
Git:


File Edit View Navigate Code Refactor Run Tools VCS Window Help
tms-z21 test.py
test (1)
Git:
test.py
1 a = 3 a: 3
2 print(a)
3
4 b = 4
5
6 print(b)
7
1: Project
Remote Host
SciView
Database
Key Promoter X
7: Structure
2: Favorites
Debug: test (1)
Debugger
Console
Frames
Variables
Watches
Mai.....
<module>, test.py:4
a = {int} 3
Special Variables
No watches
5: Debug
6: TODO
9: Version Control
Database Changes
Terminal
Python Console
Event Log
Python Debugger Extension Available: Cy... (moments ago)
4:1
LF
UTF-8
4 spaces
Git: cw14
Python 3.7

Взаимодействие с файловой системой

```
import os

file_path = os.path.realpath( __file__)
dir_name = os.path.dirname(file_path)
os.mkdir('path to dir')
```

os - библиотека для работы с операционной системой.

Задание 14.05

Создать скрипт, который принимает имя папки и создает ее рядом со скриптом

Задание 14.06

Дописать скрипт. Программа принимает имя папки и имя файла. Создает папку и создает в ней файл.

Задание 14.07

Дописать скрипт. Программа принимает имя папки и имя файла с расширением. Создает папку и создает в ней файл. Если расширение файла py - записывает в файл следующее:

```
def main():
    pass



if __name__ == '__main__':
    main()
```